Amazon Top 50 Bestselling Books

Machine Learning and Data Analysis

Brocchi Martina - Ferrari Luca

Introduction

The Amazon Top 50 Bestselling Books 2009-2019 Dataset contains **550 books**, that have been categorized into fiction and non-fiction genres using Goodreads. The analysis of this Dataset will allow us have a deep understanding of the book market trends over the past decade.

The data is described through **7 features**: name of the book, author of the book, user rating, number of reviews, the price of the book, the year(s) it ranked on the bestseller, and whether it is a fiction or non-fiction book.

Preliminary Actions

Before starting our analysis, we performed some preliminary actions on the dataset. In particular:

- We checked for **missing values** (there were none)
- We corrected **misspellings** that led to duplicates in authors' names
  - E.g. J. K. Rowling and J.K. Rowling
- We mapped the **genre of the books** into a **quantitative feature**
- We checked for **duplicated books**

Duplicated Books

To better analyze the books, we **removed the year feature** to manage only once the books duplicated in the dataset for different years.

	Name	Author	User Rating	Reviews	Price	Year	Genre
546	You Are A Badass: How To Stop Doubting Your Gr...	Jen Sincero	4.7	14331	8	2016	Non Fiction
547	You Are A Badass: How To Stop Doubting Your Gr...	Jen Sincero	4.7	14331	8	2017	Non Fiction
548	You Are A Badass: How To Stop Doubting Your Gr...	Jen Sincero	4.7	14331	8	2018	Non Fiction
549	You Are A Badass: How To Stop Doubting Your Gr...	Jen Sincero	4.7	14331	8	2019	Non Fiction

Duplicated Books

After removing duplicates caused by the year, we still had some **books duplicated for other values**. For example, the book “The Help” had a different price in different years.

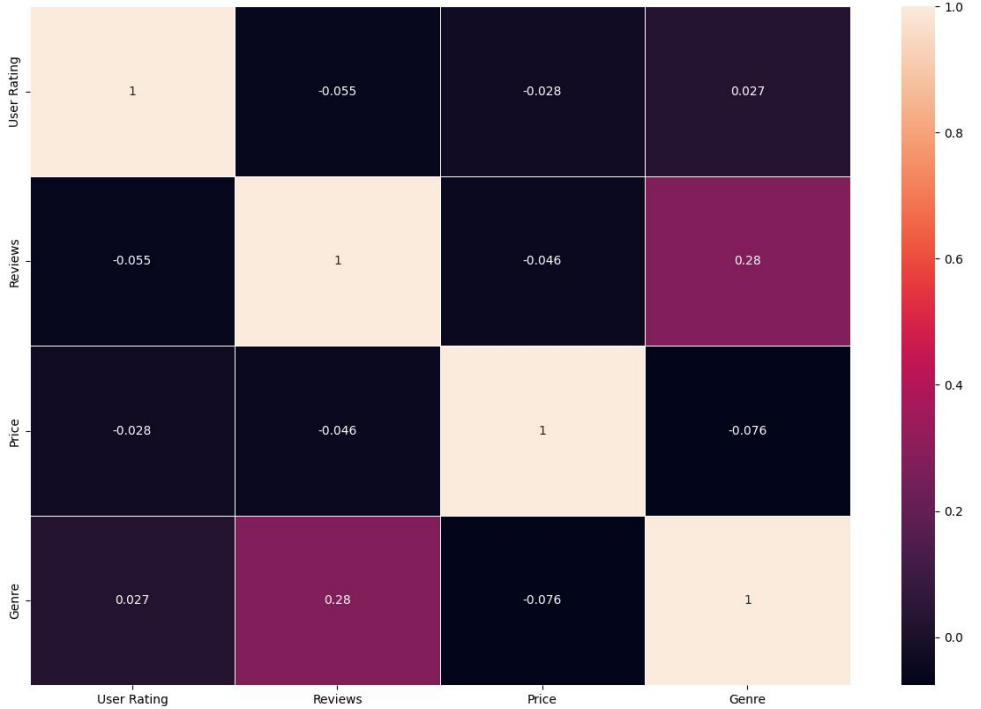
	Name	Author	User Rating	Reviews	Price	Genre
402	The Help	Kathryn Stockett	4.8	13871	6	Fiction
404	The Help	Kathryn Stockett	4.8	13871	8	Fiction
405	The Help	Kathryn Stockett	4.8	13871	7	Fiction

In those cases, we **kept the last entry** and we discarded the others. After these operations, the dataset contained **350 different books** written by **246 authors**.

Data Analysis

Correlation

Unfortunately, none of the features of our dataset were strongly correlated with each other, as we can see from the **correlation heatmap**, so we had to conduct our analysis looking for other properties.

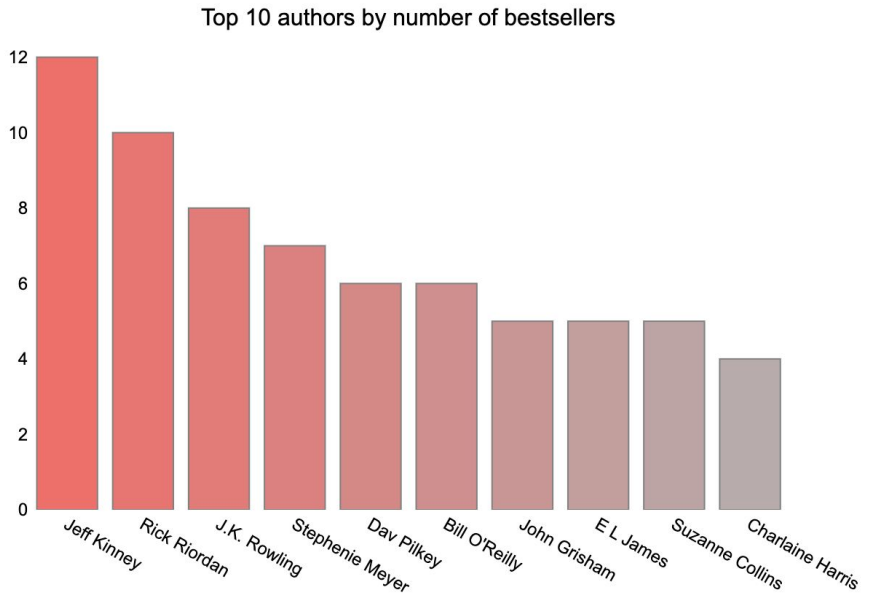


Who are the most popular authors?

Top 10 authors by number of bestsellers

To find the most popular authors we first looked for the authors who wrote the highest number of bestsellers, i.e. the authors with most books in the top 50 charts.

We may identify these authors as the most popular ones, however, we should also take into account other factors, such as their user rating.



Top 10 authors by user rating

The table shows the 10 authors with the highest average user rating. We can observe how **Dav Pilkey** is the only author who appears in both charts: he's the 9th author by average user rating and the 5th author by number of best sellers, thus, he might be the most popular author.

However, if we look at the average number of reviews, we can see that he has few reviews with respect to other authors (e.g. **Sarah Young**)

Author	User Rating	Reviews
Sarah Young	4.9	19576
Eric Carle	4.9	19546
Bill Martin Jr.	4.9	14344
Jill Twiss	4.9	11881
Nathan W. Pyle	4.9	9382
Emily Winfield Martin	4.9	8842
Chip Gaines	4.9	7861
Dav Pilkey	4.9	7376.833333333333
Sherri Duskey Rinker	4.9	7038
Lin-Manuel Miranda	4.9	5867

Top 10 authors by average number of reviews

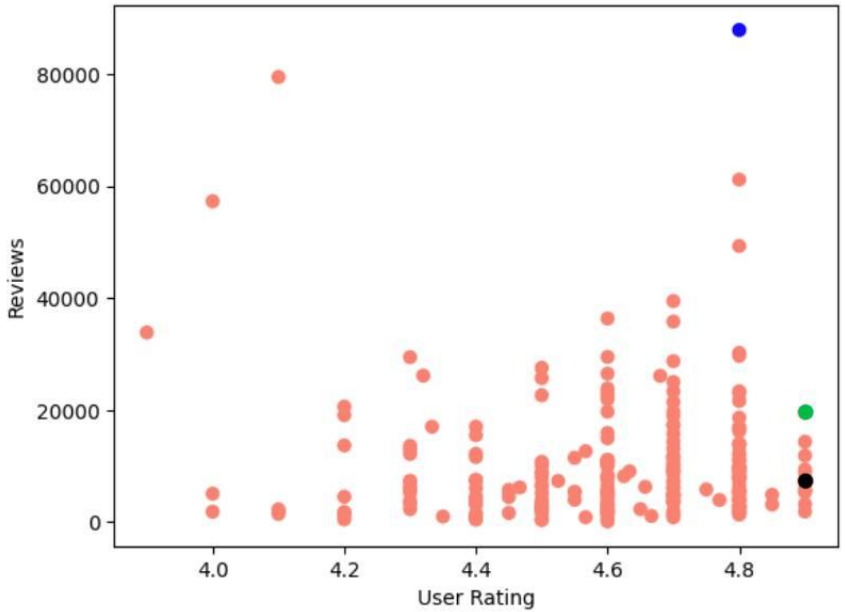
Sorting the table by the average number of reviews of the authors we can observe how the ranking completely changed: **Delia Owens** is the author with most reviews (on average) and has a pretty high user rating (4.8 out of 5).

Author	User Rating	Reviews
Delia Owens	4.8	87841
Paula Hawkins	4.1	79446
Michelle Obama	4.8	61133
Gillian Flynn	4	57271
Kristin Hannah	4.8	49288
Andy Weir	4.7	39459
Anthony Doerr	4.6	36348
Paulo Coelho	4.7	35799
Donna Tartt	3.9	33844
Craig Smith	4.8	30183

Most popular authors

To better understand the relationship between the user rating and the reviews of the most popular authors, we produced a scatter plot. In particular:

- **Delia Owens**: highest average number of reviews - user rating 4.8
- **Sarah Young**: highest user rating - highest average number of reviews
- **Dav Pilkey**: highest user rating - 6 bestsellers

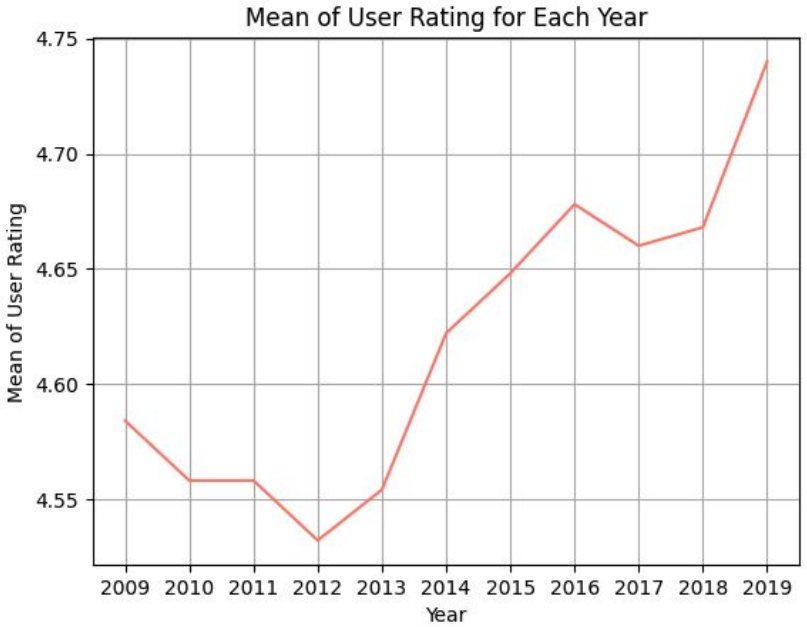


Trends over the years

User rating

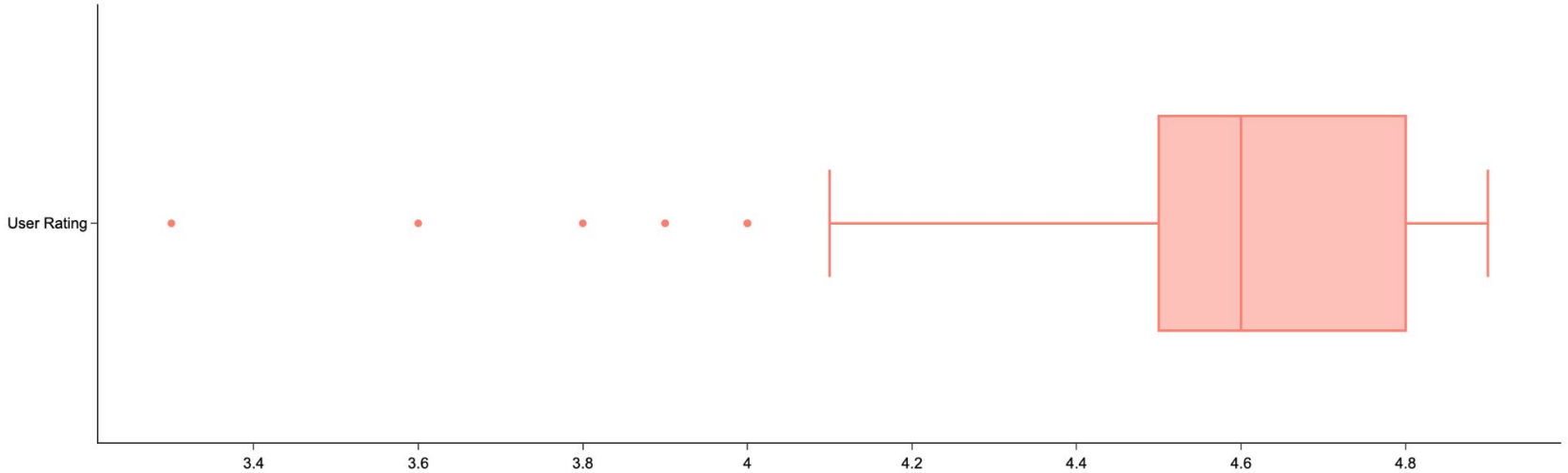
From 2009 to 2012 there was a decrease of the mean user rating, with the lowest peak registered in 2012. There are several possible reasons, one of which could be a change in the Amazon rating system that may have facilitated (negative) ratings.

However, the average user score has generally increased over the years, particularly since 2012, with its highest peak in 2019.



User rating box-plot

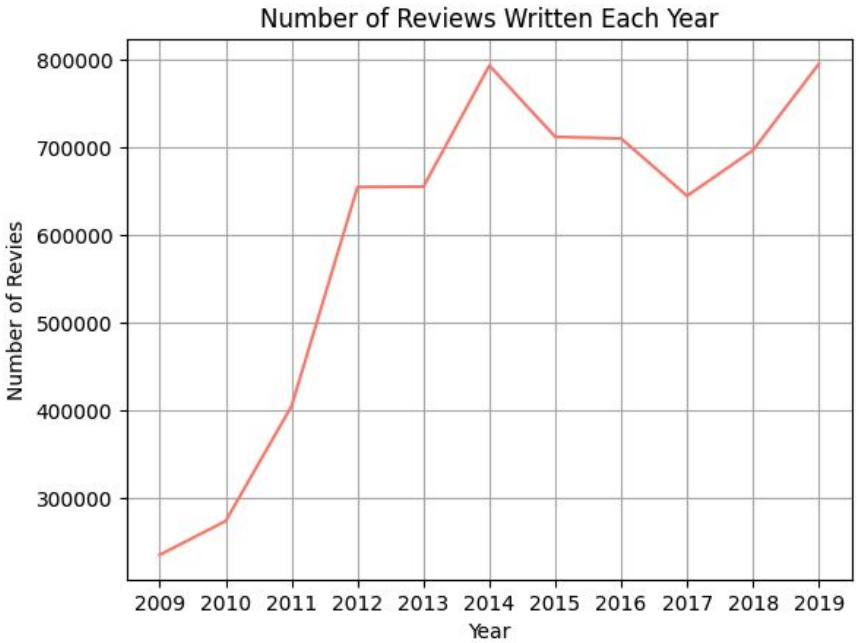
As we can see from the box-plot, the user rating is not distributed normally: the average book rating is 4.6, and there are outliers in the data (there is a small number of books in the data below the 4.1 rating)



Reviews

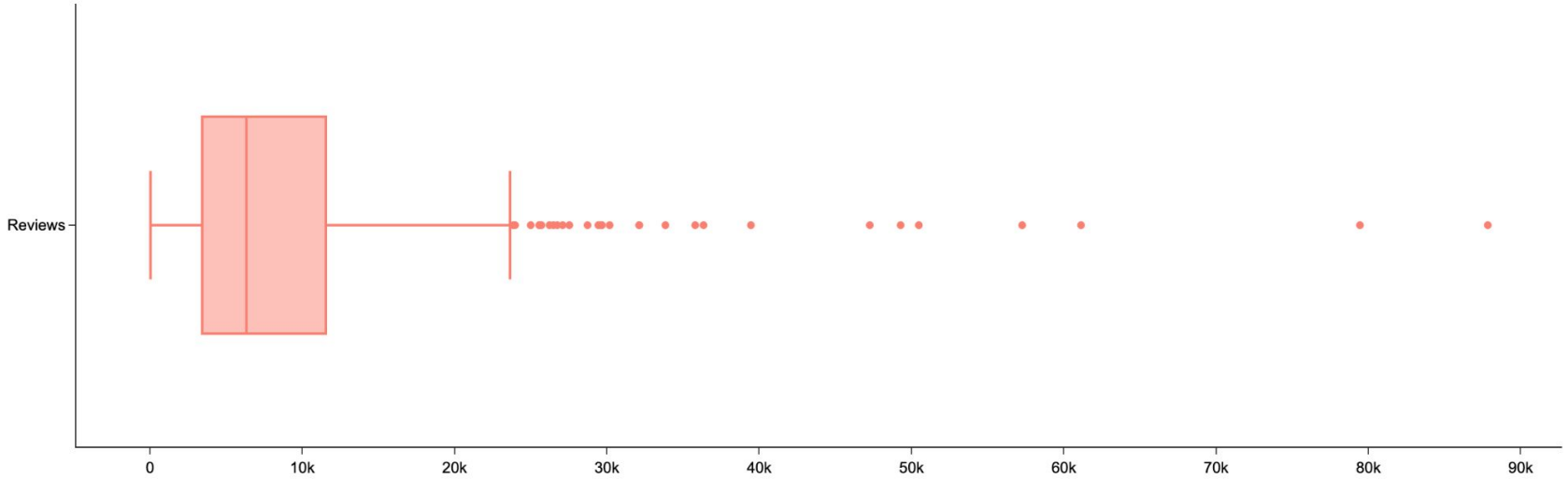
We can observe a huge increment in the number of reviews from 2009 to 2012. Interestingly, this is consistent with decreasing ratings, which supports our previous thesis: Amazon may have facilitated the rating system.

The number of reviews remained fairly stable subsequently, with two peaks of growth in 2014 and 2019.



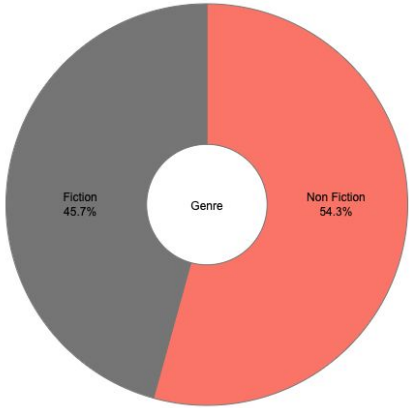
Reviews box-plot

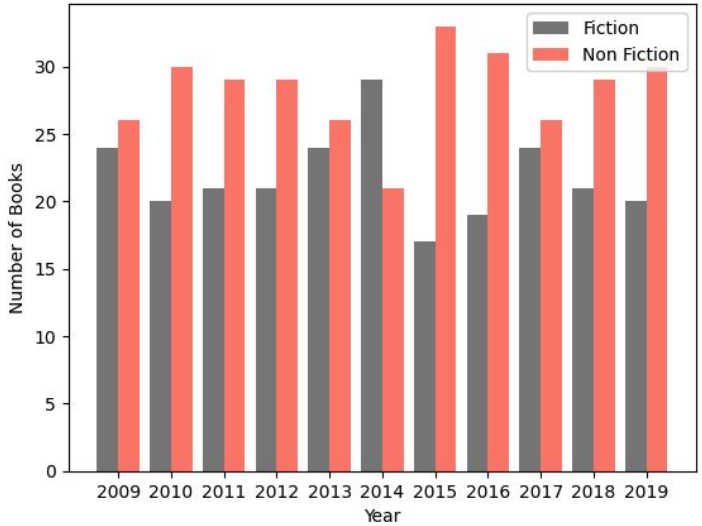
Reviews aren’t distributed normally too: the data has a wide range, and there are outliers (there is a small number of books receiving reviews well above the 75 percentile).



Genre

The most popular genre is non fiction: in particular, non fiction books were the majority of bestsellers every year, with the exception of 2014.





Machine Learning

Regression

For the machine learning analysis we first decided to solve a regression problem, **predict the user rating of a book**, using the features:

- Reviews
- Price
- Year
- Genre

The methods that we tested and compared are **Decision Tree** and **K-Nearest Neighbors.**

Preprocessing

Before proceeding with the application of the methods, we had to transform our Genre `[Fiction, Non Fiction]` into quantitative features `[1,0]`, using the `fit_transform()` method (`sklearn.preprocessing.LabelEncoder`).

After that, we divided the Training set from the Testing set, assigning 33% of our data to the Testing set.

Decision Tree
Parameter Tuning

To find the best values for the Decision Tree parameters, we used the `GridSearchCV`, obtaining the following result

`max_depth`	3
`min_samples_leaf`	1
`min_samples_split`	5

Decision Tree

Best parameters

After that, we tested the Decision Tree algorithm using the best parameters found with the GridSearchCV, obtaining the following results

	Current Values	Predicted Values
195	4.6	4.653226
79	4.7	4.729167
480	4.9	4.753731
109	4.2	4.753731
522	4.8	4.653226
..	...	...
113	4.7	4.753731
304	4.0	4.560417
173	4.8	4.533333
362	4.8	4.753731
208	4.6	4.560417

RMSE	MAE	R2
0.25	0.18	-0.02

Decision Tree

Default parameters

To better understand the results we obtained, we compared them with the ones obtainable using the default hyperparams

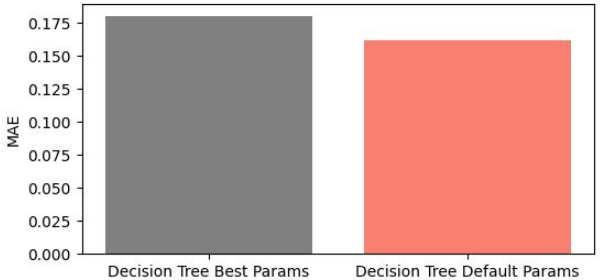
`max_depth`	None
`min_samples_leaf`	1
`min_samples_split`	2

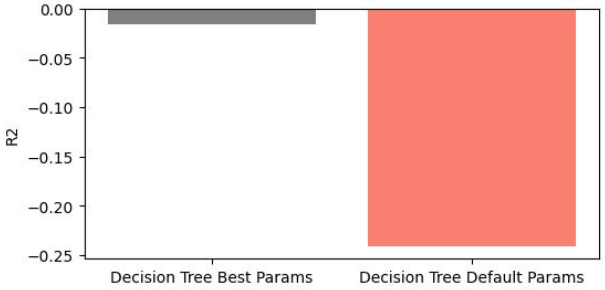
RMSE	MAE	R2
0.27	0.16	-0.25

	Current Values	Predicted Values
195	4.6	4.5
79	4.7	4.8
480	4.9	4.9
109	4.2	4.3
522	4.8	4.6
..	...	...
113	4.7	4.7
304	4.0	4.6
173	4.8	4.0
362	4.8	4.9
208	4.6	4.5

Decision Tree

Best vs Default Parameters





	MAE	RMSE	R2
Default parameters	0.16	0.27	- 0.25
Best parameters	0.18	0.25	- 0.02

K-Nearest Neighbors
Parameter Tuning

For the KNN method, we started in the same way, using the `GridSearchCV` to find the best values for the parameters.

`n_neighbors`	15
`metric`	‘manhattan’
`weights`	‘distance’

K-Nearest Neighbors

Best parameters

After that, we tested KNN with the best parameter obtained from the `GridSearchCV`, obtaining the following results

	Current Values	Predicted Values
195	4.6	4.564542
79	4.7	4.672500
480	4.9	4.896414
109	4.2	4.555653
522	4.8	4.541984
..	...	...
113	4.7	4.700655
304	4.0	4.039865
173	4.8	4.617055
362	4.8	4.783966
208	4.6	4.655446

RMSE	MAE	R2
0.20	0.12	0.33

K-Nearest Neighbors
Default parameters

RMSE	MAE	R2
0.24	0.12	0.33

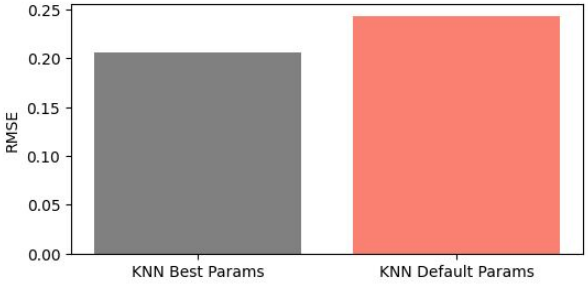
Also in this case, we tested the method with the default parameters

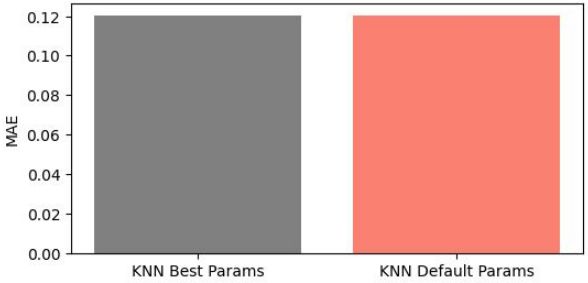
`n_neighbors`	5
`metric`	‘minkowski’
`weights`	‘uniform’

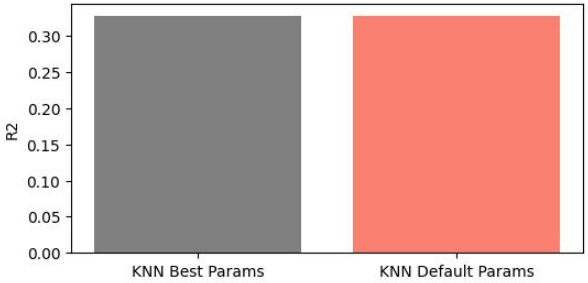
	Current Values	Predicted Values
195	4.6	4.50
79	4.7	4.64
480	4.9	4.90
109	4.2	4.54
522	4.8	4.54
..	...	...
113	4.7	4.72
304	4.0	4.16
173	4.8	4.62
362	4.8	4.78
208	4.6	4.72

K-Nearest Neighbors

Best vs Default Parameters





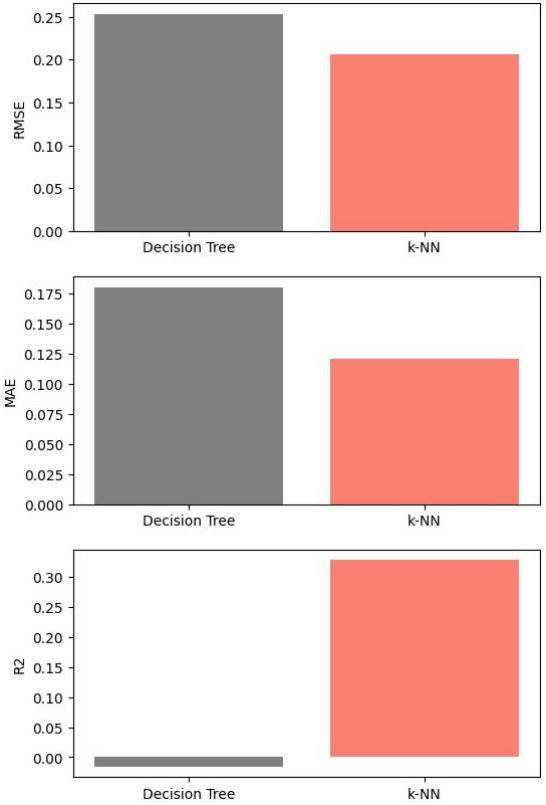


	MAE	RMSE	R2
Default parameters	0.12	0.24	0.33
Best parameters	0.12	0.20	0.33

Decision Tree vs K-Nearest Neighbors

Comparing the results we obtained testing the two methods with the best parameters, we can easily see how **KNN** has **better performances** than Decision Tree, outperforming it on all the evaluation metrics.

However, even with KNN, we obtain a **decent but not optimal result**. As both algorithms being tested are also suitable for classification, we tried to simplify the problem under analysis by treating it as a classification task.



Machine Learning

Classification

For the second part of machine learning analysis we decided to solve a classification problem, **predict if the user rating of a book will be higher or lower than the average**, using the features:

- Reviews
- Price
- Year
- Genre

We tested and compared the same methods, **Decision Tree** and **K-Nearest Neighbors.**

Preprocessing

Before proceeding with the application of the methods, we had to transform our User Rating encoding:

- the value > of average with **1**
- the value < average with **0**

After that, we divided the Training set from the Testing set, assigning 33% of our data to the Testing set.

Decision Tree
Parameter Tuning

To find the best values for the Decision Tree parameters, we used the `GridSearchCV`, obtaining the following result

`max_depth`	5
`min_samples_leaf`	4
`min_samples_split`	2

Decision Tree

Best parameters

After that, we tested the Decision Tree algorithm using the best parameters found with the GridSearchCV, obtaining the following results

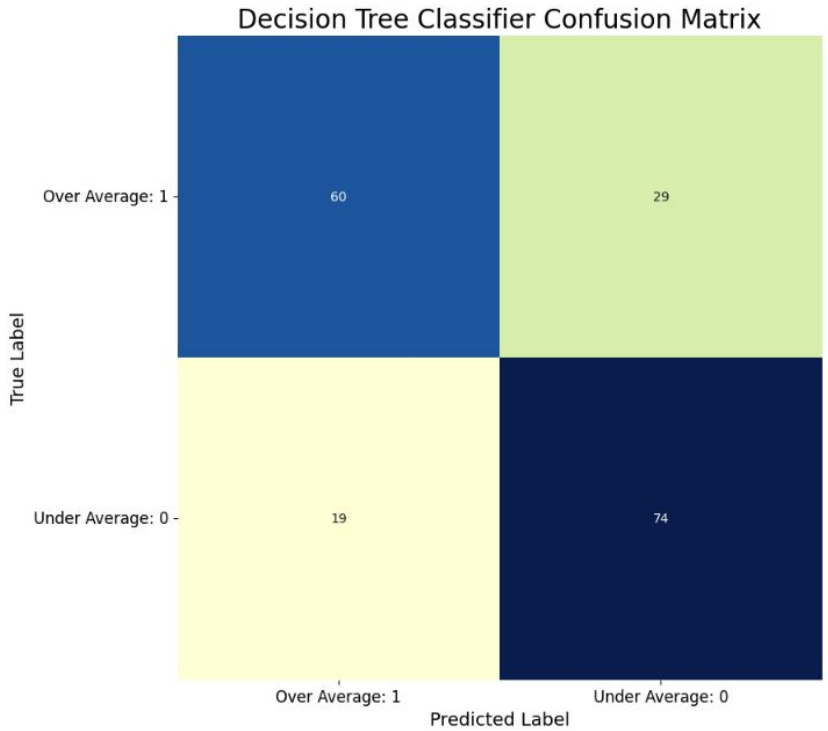
	Current Values	Predicted Values
195	0	0
79	1	0
480	1	1
109	0	0
522	1	0
..	...	...
113	1	1
304	0	0
173	1	0
362	1	1
208	0	0

Accuracy	Precision	Recall	AUC
0.73	0.71	0.79	0.73

Decision Tree
Confusion Matrix

The results described by the confusion matrix, on which the metrics were computed, are:

- True Positive (TP) = 60
- False Negative (FN) = 29
- False Positive (FP) = 19
- True Negative (TN) = 74



Decision Tree

Default parameters

To better understand the results we obtained, we compared them with the ones obtainable using the default hyperparams

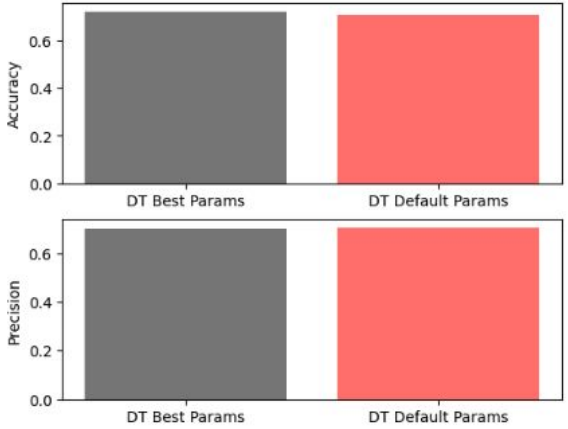
max_depth	None
min_samples_leaf	1
min_samples_split	2

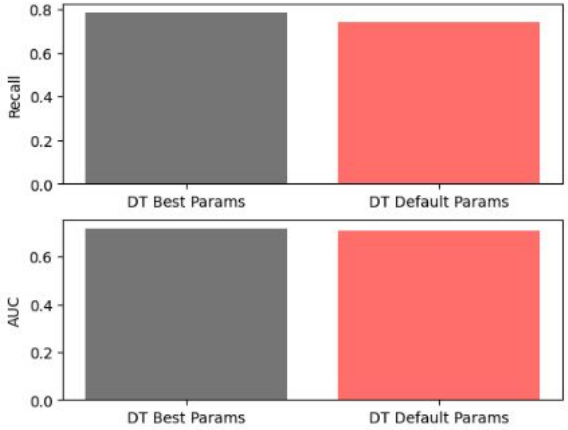
Accuracy	Precision	Recall	AUC
0.70	0.70	0.74	0.70

	Current Values	Predicted Values
195	0	0
79	1	0
480	1	1
109	0	0
522	1	0
..	...	...
113	1	1
304	0	0
173	1	1
362	1	1
208	0	0

Decision Tree

Best vs Default Parameters





	Accuracy	Precision	Recall	AUC
Default parameters	0.70	0.70	0.74	0.70
Best parameters	**0.73**	**0.71**	**0.79**	**0.73**

K-Nearest Neighbors
Parameter Tuning

For the KNN method, we started in the same way, using the `GridSearchCV` to find the best values for the parameters.

`n_neighbors`	7
`metric`	‘manhattan’
`weights`	‘distance’

K-Nearest Neighbors

Best parameters

After that, we tested the KNN algorithm using the best parameters found with the GridSearchCV, obtaining the following results:

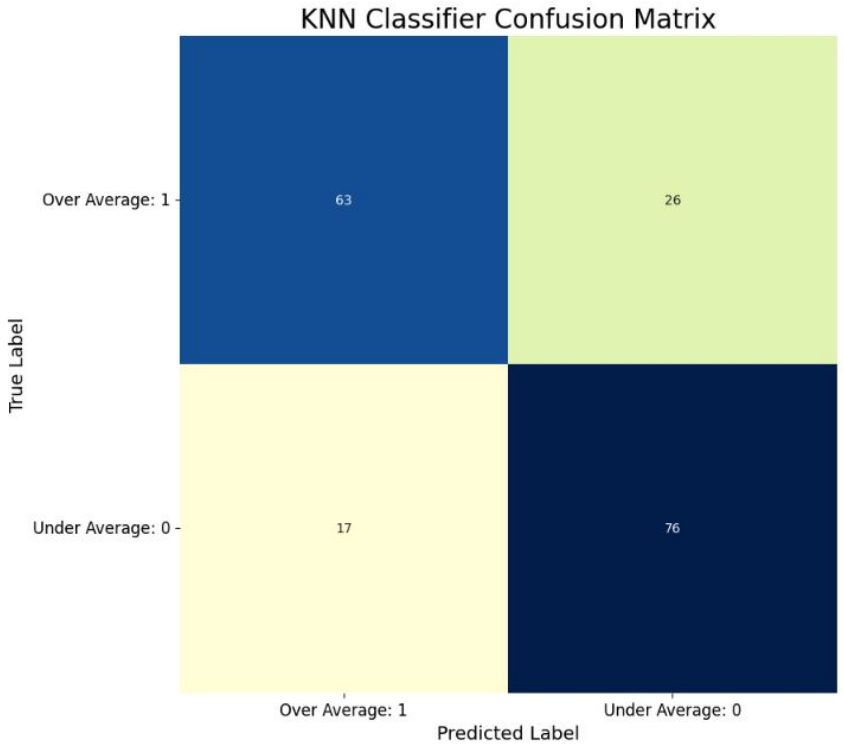
	Actual Values	Predicted Values
195	0	0
79	1	1
480	1	1
109	0	0
522	1	0
..	...	...
113	1	1
304	0	0
173	1	1
362	1	1
208	0	0

Accuracy	Precision	Recall	AUC
0.73	0.74	0.81	0.76

K-Nearest Neighbors
Confusion Matrix

The results described by the confusion matrix, on which the metrics were computed, are:

- True Positive (TP) = 63
- False Negative (FN) = 26
- False Positive (FP) = 17
- True Negative (TN) = 76



K-Nearest Neighbors

Default parameters

To better understand the results we obtained, we compared them with the ones obtainable using the default hyperparams

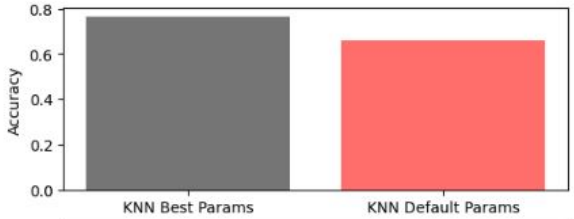
Accuracy	Precision	Recall	AUC
0.65	0.64	0.73	0.65

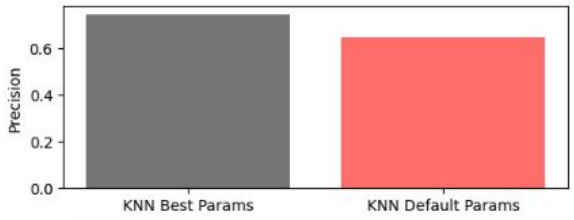
`n_neighbors`	5
`metric`	‘minkowski’
`weights`	‘uniform’

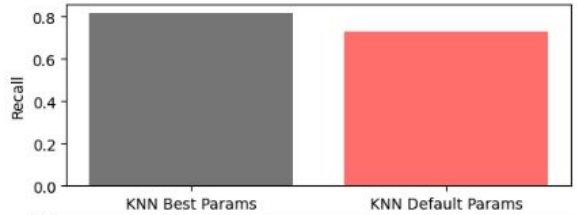
	Actual Values	Predicted Values
195	0	0
79	1	1
480	1	1
109	0	0
522	1	0
..	...	...
113	1	1
304	0	0
173	1	1
362	1	1
208	0	1

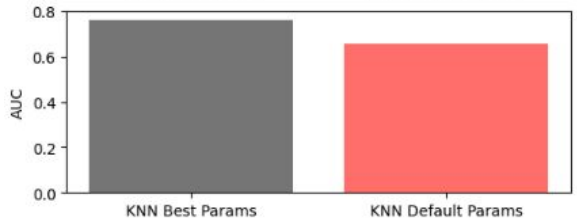
K-Nearest Neighbors

Best vs Default Parameters





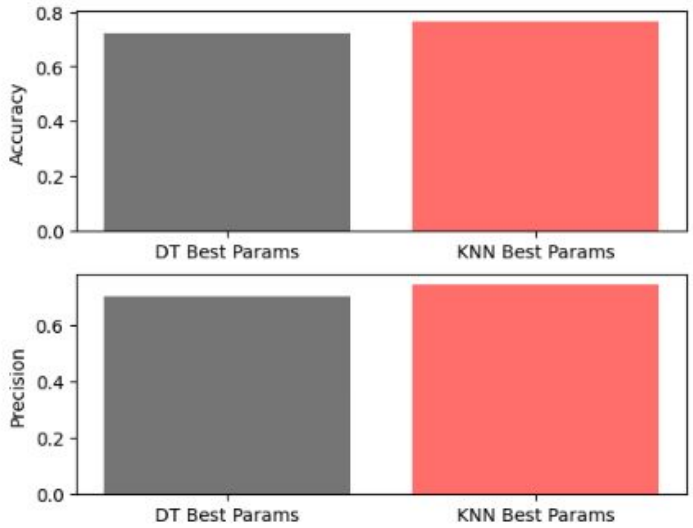


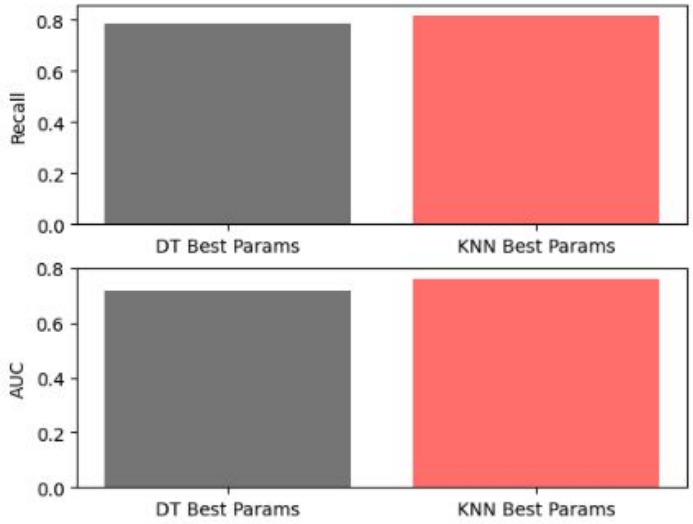


	Accuracy	Precision	Recall	AUC
Default parameters	0.65	0.64	0.73	0.65
Best parameters	**0.73**	**0.74**	**0.81**	**0.76**

Decision Tree vs K-Nearest Neighbors

Comparing the results we obtained testing the two methods with the best parameters, we can see even if by a little that **KNN** has **better performances** than Decision Tree.





Conclusions

In conclusion, we can say that efficient parameter tuning is crucial for machine learning algorithms, especially, in our case, for K-Nearest Neighbors (KNN).

Additionally, we have observed that classification performs better than regression for the selected dataset and the selected features, where regression models were found to be inefficient, while classifiers proved to be effective.

Lastly, in our comparison of KNN and Decision Tree, KNN emerged as the superior algorithm. Therefore, when dealing with similar datasets and features, we recommend considering parameter tuning and choosing a classification model such as KNN to achieve accurate and efficient results.